Universidad San Francisco de Quito USFQ

Colegio de Ciencias e Ingenierías

Estudio de la eficiencia de algoritmos de reconocimiento de números, después de una reducción del espacio de características mediante el cálculo de testores típicos

Proyecto de Investigación

Kuntur Mallku Muenala Terán

Matemáticas

Trabajo de titulación presentado como requisito para la obtención del título

de

Licenciado en Matemáticas

Quito, 18 de mayo de 2018

Universidad San Francisco de Quito USFQ

Colegio de Ciencias e Ingenierías

Hoja de calificación
de Trabajo de titulación

Estudio de la eficiencia de algoritmos de reconocimiento de números, después de una reducción del espacio de características mediante el cálculo de testores típicos

Kuntur Mallku Muenala Terán

Calificación:

Nombre del Profesor, Título académico: Julio Ibarra, MSc.

Firma del profesor: _____

Quito, 18 de mayo de 2018

Derechos de Autor

Por medio del presente documento certifico que he leído todas las Políticas y Manuales de la Universidad San Francisco de Quito USFQ, incluyendo la Política de Propiedad Intelectual USFQ, y estoy de acuerdo con su contenido, por lo que los derechos de propiedad intelectual del presente trabajo quedan sujetos a lo dispuesto en esas Políticas.

Asi mismo, autorizo a la USFQ para que realice la digitalización y publicación de este trabajo en el repositorio virtual, de conformidad a lo dispuesto en el Art. 144 de la Ley Orgánica de Educación Superior.

Firma del Estudiante: _____

Nombres y Apellidos: Kuntur Mallku Muenala Terán

Código: 00107802

Cédula de Identidad: 1003667209

Lugar y Fecha: Quito, 18 de mayo de 2018

Resumen

En el presente trabajo se analiza las predicciones hechas por los modelos logístico y multilogístico usando testores típicos calculados mediante el algoritmo de Takeaki Uno obtenido en Hypergraph Dualization Repository [1], a partir del conjunto de características de imágenes binarias de una base de datos escritas a mano obtenidas de la base de datos Semeion Handwritten Digit Data Set obtenido en Machine Learning Repository [2]. Comprando la eficiencia de las predicciones obtenidos con todo el conjunto de características y el conjunto reducido de características obtenidos a partir de el cálculo de testores típicos, la implementación de los modelos y algoritmos se los realiza en los lenguajes de programación Julia y Matlab.

Palabras claves: modelo logístico, modelo multilogístico, testores típicos, binario, conjunto de características, base de datos, algoritmos.

Abstract

In the present work, the predictions made by the logistic and multilogistic models are analyzed using typical tweeters calculated using the algorithm of Takeaki Uno obtained in Hypergraph Dualization Repository [1], from the set of binary image characteristics of a database Handwritten Digit Data Set obtained from Machine Learning Repository [2]. Buying the efficiency of the predictions obtained with the whole set of characteristics and the reduced set of characteristics obtained from the calculation of typical testers, the implementation of the models and algorithms are carried out in the Julia and Matlab programming languages.

Keywords: logistic model, multilogistic model, typical testers, binary, set of characteristics, database, algorithms.

Índice

Índice de tablas

Índice de figuras

1. Introducción

El presente trabajo estudia una aplicación de la teoría de testores típicos. la idea es reducir el espacio de características para usarlas como insumo en modelos estadísticos para reconocer a que digito corresponde una imagen binaria. Se explica el uso de los modelos de regresión logístico y multilogístico en las predicciones.

Con la ayuda de los lenguajes de programación Julia y Matlab se implementan los scripts (programas) necesarios para el cálculo de testores típicos y de los modelos estadísticos mencionados, finalmente se presentan los resultados en porcentajes.

Se usa una base de datos extraída de Semeion Research Center of Sciences of Communication [2], esta base de datos consiste en una muestra de imágenes de números. Alrededor de 80 personas escribieron los dígitos del 0 al 9 en una caja rectangular de 16×16 en una escala de 256 valores. De esta manera de digitalizaron 1593 dígitos escritos a mano, donde cada píxel de cada imagen se escaló un valor booleano (1/0). Cada persona escribió dos veces el mismo digito, reto fue escribir la primera vez de manera normal (tratando de escribir todos los dígitos con precisión) y la segunda de una manera rápida (perdiendo la precisión) [2].

2. Imagenes y Pixeles

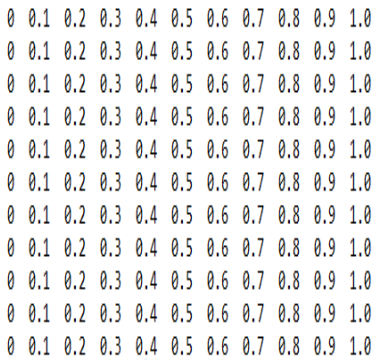
La mayoría de las imágenes por computadora son representadas como matrices rectangulares de pixeles, los pixeles es la parte elemental de cada imagen y estos guardan valores sobre la información de la imagen [3].

2.1. Imagenes representadas con matrices binarias

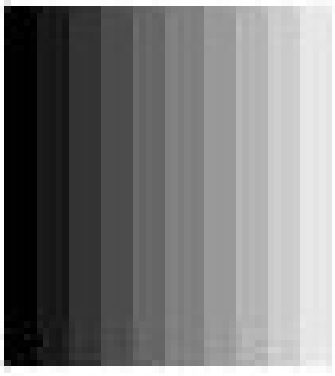
Son imágenes de matrices de dos dimensiones donde cada pixel tiene un valor entre 0 a 1, en la escala de los números reales, 0 es negro y 1 blanco, como se muestra en la siguiente figura.

(a) Matriz de la imagen

(b) Imagen computarisada



(c) Matriz Escala de grises



(d) Imagen Escala de grises

Figura 1: diferencia entre imegen binaria y escala de grises

2.2. Imagenes RGB a color

Existen varios formatos de imágenes a color, pero uno de los más conocidos es las imágenes RGB (red, green, blue), estas imágenes son matrices tridimensionales, donde las dos primeras dimensiones son las dimensiones de la imagen y la tercera guarda el valor del pixel en coordenadas RGB, es decir se tiene tres imágenes una en escala de rojo, otra en escala de verde y la tercera en escala de azul, y al juntarlas se obtiene la imagen a color original, como se presenta en la siguiente figura.

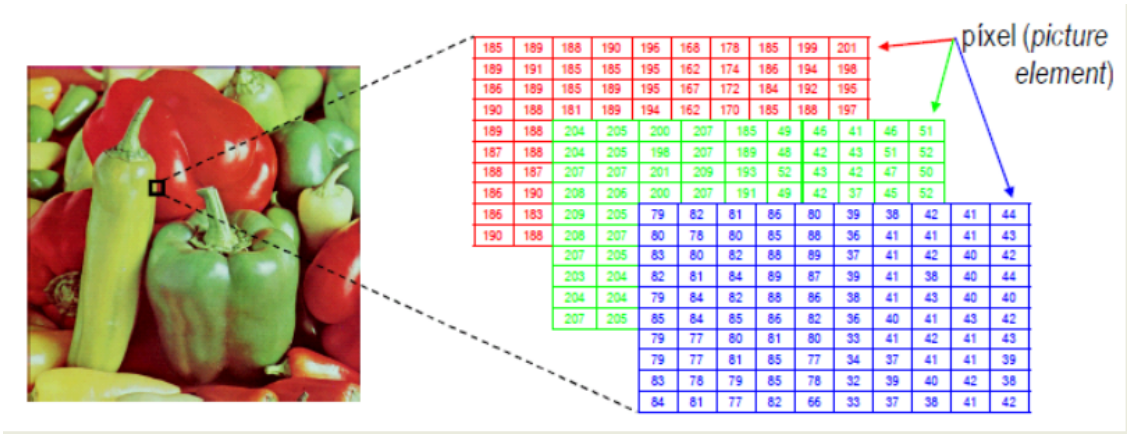


Figura 2: Ejemplo de imagen en RGB

3. Testores Típicos

Los testores típicos juegan un rol muy importante en los problemas de reconocimiento supervisado de patrones, al usar el enfoque lógico combinatorio. Un testor es una colección de características que discriminan las descripciones de objetos pertenecientes a diferentes clases, y es mínimo en el orden parcial determinado por la colección de estos conjuntos. Al dar una solución en los problemas de reconocimiento de patrones, los testores típicos juegan un papel importante en la selección de variables, permiten detectar si es existen conjuntos de variables con menor cardinalidad que mantiene la capacidad de discriminación en las clases propuestas.

En el enfoque lógico combinatorio, la información de reconocimiento supervisado de patrones puede ser reducida a una matriz, a esta matriz se le da el nombre de matriz M. Los testores típicos se buscan entre todos los posibles subconjuntos de columnas de esta matriz M, estas columnas guardan información sobre las características de cada objeto. Este trabajo se basa en el documento "Generación de matrices para evaluar el desempeño de estrategias de búsqueda de testores típicos" de Eduardo Alba-Cabrera y Roberto Santana [4]. Los conceptos, definiciones y teoremas son propios de originarios de dichos autores.

3.1. Conceptos de Testor y testor típico

Sea U una colección de objetos, estos objetos se describen mediante un conjunto de n características y están agrupados en l clases. Se compara característica a característica de cada para de objetos pertenecientes a diferentes clases, de esta manera se obtiene la matriz $M = [m_{ij}]_{p\times n}$ donde $m_{ij} \in \{0, 1\}$ y p es el número de pares. $m_{ij} = 1$ significa que los objetos del par denotado por i son similares en la característica j, mientras que $m_{ij} = 0$ indica que los objetos del par denotado por i son diferentes. Esta comparación entre características de cada par de objetos forma la matriz M de similitud, mientras que su viceversa es decir $m_{ij} = 0$ para similitud entre características de cada par de objetos y $m_{ij} = 1$ para diferentes característica de cada par de objetos, forman la matriz M de disimilitud. En el presente trabajo se trabajará con la matriz M de disimilitud.

Sea $I = \{i_1, ..., i_p\}$ el conjunto de las filas de M y $J = \{j_1, ..., j_n\}$ el conjunto de las columnas (características) de M. Sea $T \subseteq J$ y $M_{/T}$ la matriz obtenida de M al eliminar todas las columnas que no pertenecen al conjunto T, es decir $M_{/T}$ es la matriz asociada al conjunto T al representarlo como matriz y no como un conjunto de columnas, la matriz $M_{/T}$ tiene dimensiones $p \times q$, donde p es el número de pares de comparación y q es el número de elementos del conjunto J.

Definición 1: Un conjunto $T = j_{k_1}, ..., j_{k_s} \subseteq J$ es un testor de M si no existe ninguna fila de ceros en $M_{/T}$, donde $\{j_{k_s}\}$ son subíndices de $\{j_n\}$.

Definición 2: La característica $j_{k_r} \in T$ es típica con respecto a T y M si $\exists h$, $h \in \{1, ..., p\}$ tal que $a_{i_h j_{k_r}} = 1$ y para $s > 1$ $a_{i_h j_{k_t}} = 0$, $\forall t$, $t \in \{1, ..., s\}$ $t \neq r$.

Definición 3: Un conjunto T tiene la propiedad de tipicidad con respecto a una matriz M si todas las características en T son típicas con respecto a T y M.

Definición 4: Un conjunto $T = j_{k_1}, ..., j_{k_s} \subseteq J$ se denomina testor típico de M si es un testor y tiene la propiedad de tipicidad con respecto a M.

Proposición 1: Un conjunto $T = j_{k_1}, ..., j_{k_s} \subseteq J$ tiene la propiedad de tipicidad con respecto a la matriz M si y sólo si se puede obtener una matriz identidad en $M_{/T}$, eliminando e intercambiando

algunas filas.

Sea a y b dos filas de M.

Definición 5: Decimos que a es menor que b ($a < b$) si $\forall i\ a_i \leq b_i$ y $\exists j$ tal que $a_j \neq b_j$, donde $\{a_i\}$ y $\{b_i\}$ son elementos de las filas a y b respectivamente, e i y j son índices del número de columnas de M, $i = \{1, .., q\}$.

Definición 6: a es una fila básica de M si no existe otra fila menor que a en M.

Definición 7: La matriz básica de M es la matriz M' que sólo contiene todas las filas básicas de M.

La siguiente proposición es una caracterización de la matriz básica.

Proposición 2: M' es una matriz básica si y sólo si para dos filas a y b cualesquiera, a, $b \in M'$ existen dos columnas i y j tales que $a_i = b_j = 1$ y $a_j = b_i = 0$.

Se dice que todas las filas básicas de una matriz básica son filas incomparables entre si.

Dada una matriz A, denotamos $\Psi^*(A)$ como el conjunto de todos los testores típicos de A.

Proposición 3: $\Psi^*(M) = \Psi^*(M')$.

De acuerdo con la proposición 3, es conveniente encontrar la matriz M' y luego calcular el conjunto $\Psi^*(M')$. Debido a que M' tiene menor o igual número de filas que M, eso ayuda en la eficiencia de los algoritmos para hallar todos los testores típicos de M.

4. Modelos estadísticos de regresión

Un modelo de regresión es un modelo matemático que busca determinar la relación entre una variable dependiente o respuesta con respecto a otras variables independientes o explicativas [5], normalmente se representa a la variable dependiente (respuesta) como Y y a la variable independiente (explicativa) como X. Existen varios modelos de regresión los cuales tienen muchas aplicaciones en diferentes áreas académicas como el campo de las ciencias sociales hasta la biología. Sin embargo, en el estudio presente solo se estudiará la regresión logística y la regresión multilogísitica [5].

4.1. Regresión logística

En algunas situaciones de los modelos de regresión, la variable respuesta y_i solo tiene dos posibles resultados 0 o 1, por ejemplo, si la presión de la sangre es alta o baja, si el cáncer en una persona sigue desarrollándose o no, y otros ejemplos más [6]. En estos casos la variable y_i se le asigna uno de los valores 0 o 1, como un sí o no, para predecir la probabilidad p_i del resultado sobre la base de datos de uno o varios x_i.

El modelo de regresión logística es dado por la función:

$$p_i = E(y_i) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_{1,i} + ... + \beta_n x_{n,i})}} \tag{1}$$

El modelo puede ser linealizado por la simple transformación:

$$\ln\left(\frac{p_i}{1 - p_i}\right) = \beta_0 + \beta_1 x_{1,i} + ... + \beta_n x_{n,i} \tag{2}$$

Esta transformación se la llama *logit*.

Los coeficientes $\beta_0, ..., \beta_n$ son estimadores calculados por el método de máxima verosimilitud. Para ver como son calculados los coeficientes $\beta_0, ..., \beta_n$ por dicho método véase "Linear Models In

Statistics" de Alvin C. Rencher y G. Bruce Schaalje [6].

La regresión logística se ha extendido desde la logística binaria hasta la logística policromática, es decir modelo de regresión en el que y_i tiene varios resultados posibles [6]. Esta es la regresión multilogística presentada a continuación.

4.2. Regresión multilogística

La regresión multilogística es también denominada regresión logística multinomial por que la variable dependiente es de tipo nominal con más de dos categorías (politómica) como respuesta y también es una extensión multivariante de la regresión logística binaria presentada anteriormente [7].

El modelo de regresión multilogística es dado por las siguientes funciones:

Sea $Z_{ij} = \beta_{0,j} + \beta_{1,j}x_{i,1} + ... + \beta_{n,j}x_{i,n}$

$$p_{ij} = \frac{e^{Z_{ij}}}{1 + \sum_{k=1}^{g-1} e^{Z_{ik}}} = P[Y_i = j|x_1, x_2, ..., x_n]; j = 1, 2, .., g-1 \tag{3}$$

$$p_{ig} = \frac{1}{1 + \sum_{k=1}^{g-1} e^{Z_{ik}}} = 1 - \sum_{j=1}^{g-1} p_{ij} \tag{4}$$

Donde p_{ij} es la probabilidad del individuo i que pertenezca a la categoría j, p_{ig} es la probabilidad del individuo i que pertenezca a la categoría g, la categoría g se la denomina como categoría de referencia, de la variable con distribución multinomial Y [8].

De igual manera el modelo puede ser linealizado por:

$$\ln\left(\frac{p_{ij}}{p_{ig}}\right) = \beta_{0,j} + \beta_{1,j}x_{i,1} + ... + \beta_{n,j}x_{i,n} = Z_{ij} \tag{5}$$

Esta transformación se la llama *logit*.

Los coeficientes $\beta_{0,j}, \beta_{1,j}, ..., \beta_{n,j}$, son estimadores de la regresión multilogística, estos estimadores se los calcula por el método de máxima verosimilitud. Para estudiar más a fondo el modelo multilogístico se recomienda ver "Planteamiento del Modelo Logístico multinomial a través de la función canónica de enlace de la familia exponencial"de Osorio, D. Ospina, J. Lenis, D. [8], o libros sobre modelos estadísticos multivariables.

4.3. Calculo de estimadores β en Matlab

En este estudio se usa el software de Matlab para calcular los coeficientes β de cada regresión, debido a que en Matlab ya están predeterminadas las funciones de la regresión logística y multilogística, y lo que se pretende es implementarlas en el estudio.

Para la regresión logística se usa la función *glmfit* (Generalized linear model regression):

```
b = glmfit(X,y,'distr','param1','val1','param2','val2',...)
```

la función *glmfit* devuelve un vector b de $p+1\times1$, donde p es el número de las variables independientes (predicen), b es el vector de los coeficientes de estimación β, X son las variables independientes (predictoras), X es una matriz de $n \times p$, n es el número de eventos y y son las variables dependientes (respuesta), y es un vector de $n \times 1$. *distr* es la distribución de las variables dependientes: ”binomial”, ”gama”, y otros. *param1* son parámetros establecidos en matlab como ”link”, ”estdisp”, y otros. *val1* es la asignación del modelo, este puede ser ”identity”, ”log”, ”logit”, y otros [9].

Para establecer el modelo de regresión logística en la función *glmfit* se establece la distribución ”binomial”, el parámetro ” link ” y el valor ”logit”. Para ver otros tipos de modelos de regresión

lineal revise la documentación de Matlab para esta función.

Para la regresión multilogística se usa la función *mnrfit* (Multinomial logistic regression)

```
B = mnrfit(X,Y)
```

la función *mnrfit* devuelve una matriz B de $p+1 \times q-1$, donde p es el número de las variables independientes (predicen) y q la cantidad de categorías de respuesta que tiene el modelo, B es la matriz de las estimaciones de los coeficientes para la regresión multilogística de las respuestas nominales en Y sobre los predictores X. cada columna de B son los coeficientes β de una categoría respecto con la categoría de referencia. Matlab tiene varias categorías del modelo multilogístico, sin embargo, para el presente estudio se usará el modelo por defecto de la función *mnrfit* de matlab, que se lo denomina " nominal model " [9].

5. Desarrollo de trabajo

5.1. Calculo de todos los testores tipicos

En el presente estudio se usa una muestra de datos, obtenidos de Semeion Handwritten Digit Data set [2], esta muestra es la recopilación de 1593 dígitos escritos a mano, con la ayuda de aproximadamente 80 personas, digitalizaron cada muestra en una caja rectangular de 16×16 pixeles en escala de grises de 256 valores. Luego, cada píxel de cada imagen se asignó un valor booleano (1/0) según la imagen. Cada persona escribió en un papel los dígitos del 0 al 9 dos veces, la primera vez lo escribieron de manera normal tratando de escribir el dígito con precisión, la segunda vez lo escribieron rápido perdiendo la precisión (Brescia, 1994). un ejemplo de cada dígito se presenta en la figura (3), a continuación:

Figura 3: Muestra de todos los digitos

En la muestra *semeion.data* se tiene una matriz booleana 1593×266 de los cuales cada fila representa una imagen diferente, y las primeras 256 columnas representan los pixeles de cada imagen de 16×16 pixeles, y las últimas 10 columnas clasifican el tipo de dígito del 0 al 9 que representan las imágenes. Por lo tanto, se usa solo la submatriz 1593×256 para el análisis de cada imagen, además cada dígito en las imágenes resulta ser una clase de la muestra, por lo tanto en toda la muestra se tiene 10 diferentes clases, cada una respecto a un diferente digito.

Con la ayuda del lenguaje de Julia se analiza los datos de la muestra.

```
dataDigit = readdlm("semeion.data")
Digit = dataDigit[:,257:266]
ImgDigit = dataDigit[:,1:256]
```

La función *readdlm* lee una matriz de un archivo de texto donde cada línea da una fila, con elementos separados por el decímetro dado. Si todos los datos son numéricos, el resultado será una matriz numérica [10].

la matriz *ImgDigit* tiene la información de cada imagen, cada fila de la matriz es una imagen de un dígito de 16×16 pixeles.

la matriz *Digit* me indica a que clase pertenece cada imagen, la columna 1 representa el dígito (clase) 0, la columna 2 representa el dígito (clase 1), y así respectivamente con cada columna de *Digit*.

Ahora se necesita comparar cada imagen de la muestra con las demás imágenes de la muestra, esta comparación se la realiza con el pixel (característica) de cada imagen respecto al mismo pixel (característica) de otra imagen. En el caso de la matriz de datos que se tiene, se analiza cada valor de la misma columna en diferentes filas, a esta matriz se la llama matriz M. $M = [m_{ij}]$, donde $m_{ij} \in \{0, 1\}$, i es el número de comparación de dos filas de la matriz M, j es el número de la columna, en este caso el pixel (característica), que se compara de una imagen a otra, un ejemplo se muestra en la figura (4).

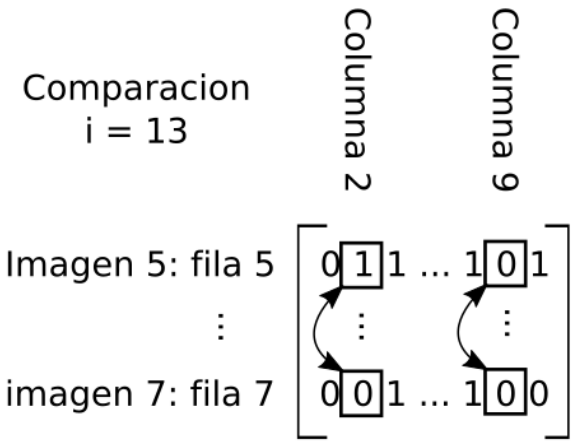


Figura 4: Comparación entre pixeles de imagenes diferentes

En nuestro estudio se escogerá la matriz M de disimilitud, definida en la sección de Testores Típicos. Como ejemplo en la figura 4 se tiene que $m_{13\ 2} = 1$ y $m_{13\ 9} = 0$, donde $i = 13$ es la comparación de la imagen 5 y 7 de la muestra.

De igual manera con ayuda de Julia, se crea un algoritmo para sacar la matriz M de disimilitud, al comparar pixel (característica) a pixel (característica) de cada par de imágenes (filas) pertenecientes a clases diferentes, el algoritmo usado es:

```
function MatrizM(A,l,nl)

# A:: la matriz de objetos con diferentes clases.
# l:: el numero de clases que hay en A.
# nl:: el numero de objetos en cada clase.

SZ = size(A)        # Las dimensiones de la matriz que se ingresa.
n = SZ[1]           # n :: filas.
m = SZ[2]           # m :: columnas.
q = l*(nl*(nl-1)/2) # q :: par de objetos comparados entre la misma clase.
```

```
p = n*(n-1)/2 - q
# p :: el total de cada par de objetos comparados, numero de filas de M.
# se resta q porque no interesa la comparacion de objetos de la misma clase.
M = ones(p,m)       # Matriz con las mismas dimensiones de la matriz M.
d = 1               # contador para cambiar la informacion de M.
for h = 1:l-1       # se repite la iteraccion por cada clase h.
    for i = 1 + (h-1)*nl:h*nl
    # recorre la primera fila hasta la ultima fila de la clase h.
        for j = 1 + h*nl:n
        # recorre las filas de la matriz A que no esten en la clase h.
          for k = 1:m
          # analiza pixel a pixel de las imagenes en comparacion.
            M[d,k] = 1*(A[i,k] .!= A[j,k])  # 1 :: diferentes, 0 :: similares.
          end
          d = d + 1
        end
    end
end
    return M        # retorna la matriz M.
end
```

Este algoritmo especifica que cada clase tiene el mismo número de objetos, esto no necesariamente tiene que ser así, sin embargo, por eficiencia del algoritmo se escoge el mismo número de objetos en cada clase. Por la variable *Digit* es fácil ver cuantas imágenes (objetos) se tiene de cada dígito (clase), esto se representa en la siguiente matriz:

$$\begin{bmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 161 & 162 & 159 & 159 & 161 & 159 & 161 & 158 & 155 & 158 \end{bmatrix}$$

Por facilidad del análisis se escogerá 150 objetos de cada clase, por lo tanto, se obtendrá una mues-

tra de 1500 imágenes y una matriz de 1500×256.

El resultado de la matriz M, dada por el algoritmo con la matriz de muestra escogida, es una matriz booleana de 1012500×256 filas y columnas, el número de filas indica que se tiene 1012500 comparaciones de diferentes clases de objetos de las 1500 imágenes en la muestra. Una vez obtenido la matriz M, esta tiene información de disimilitud de todas las imágenes de diferentes clases de la muestra.

Ahora es necesario reducir la matriz M a su matriz básica M'. Para obtener la matriz básica de M, se usan dos algoritmos en el lenguaje de Julia.

El primer algoritmo se encarga de eliminar todas las filas cero de la matriz M.

```
function del_FilZero(M)
  mn = size(M)
  # mn contiene las dimensiones de la matriz que se ingresa.
  bandera = zeros(mn[1],1)
  # crea un vector para marcar las filas de la matriz M.
  for i=1:mn[1]             # compara todas las filas.
    if sum(M[i,:]) == 0
    # si encuentra una fila cero, lo marca en bandera.
      bandera[i] = 1
    end
  end
  Vbandera = vec(bandera)   # cambio del tipo de variable, Array to Float.
  M = M[Vbandera.==0,:]     # se eliminana todas las filas marcadas.
  return M
end
```

El segundo algoritmo elimina todas las filas que no sean básicas, y mantiene todas las filas menores o básicas.

```
function MatrizB(M)

mn = size(M)
# mn contiene las dimensiones de la matriz que se ingresa.
bandera = zeros(mn[1],1)
# crea un vector para marcar filas:: 0 no es basica, 1 es basica.

for i=1:mn[1]-1             # crea un bluce para comparar dos filas.
  if bandera[i] == 0        # solo analiza las filas no marcadas.
    for j=i+1:mn[1]         # interactua con el primer for.
      if bandera[j] == 0    # solo analiza las filas no marcadas.
        cond1 = true        # se crea dos variables cond1 y cond2.
        cond2 = true
        for k=1:mn[2]       # analiza cada valor de las filas.
          if M[i,k] > M[j,k]
          # condicion 1 si fil(i) <= fil(j), cond1 cambia.
            cond1 = false
          elseif M[i,k] < M[j,k]
          # condicion 2 si fil(i) >= fil(j), cond2 cambia.
            cond2 = false
          end
          if cond1 == false && cond2 == false
          # si se cumple las dos condiciones las filas no son comparables.
            break
          end
        end
```

```
        if cond1 == true
        # si se cumple la condicion 1, se marca la fila.
          bandera[j] = 1
        elseif cond2 == true
        # si se cumple la condicion 2, se marca la fila.
          bandera[i] = 1
          break
        end
      end
    end
  end
end
vbandera = vec(bandera)
# cambio del tipo de variable, Array to Float.
M = M[vbandera.==0,:]
# se eliminan todas las filas marcadas (no son basicas).
return M
end
```

El segundo algoritmo es basado en la proposición 2, de la sección de testores típicos. Debido a que resulta más eficiente encontrar filas no comparables que una fila mayor que otra.

Al reducir la matriz M a su matriz básica M' con los algoritmos mostrados, se obtiene una matriz de 1011965×256 filas y columnas, obteniendo una reducción de 535 filas, esta matriz contiene información necesaria de la disimilitud entre imágenes de diferentes clases, por lo tanto para la matriz ya reducida se calculan todos los testores típicos de la muestra, para esto se utiliza el algoritmo *SHD (Sparsity-based Hypergraph Dualization)* sacado de Hypergraph Dualization Repository [1], un estudio de algoritmos que calculan los *minimal hitting set* que son equivalentes a los testores típicos en la teoría de Hipergrafos.

Una vez calculado $\Psi\{M'\}$ el conjunto de todos los testores típicos de M', se escoge al testor típico que tenga características más frecuentes dentro del conjunto $\Psi\{M'\}$, pues se postulará aquel testor como un buen candidato para discriminar todas las imágenes comparadas.

Sin embargo por facilidad de cálculo se halla la matriz M de solo dos dígitos 0 y 1 con 150 imágenes cada uno, esta matriz se la llamará M_1 y su matriz básica M_1', de esta manera se escogerá un $T_1 \in \Psi\{M_1'\}$ para ser usado en el modelo de regresión logística. Para la regresión multilogística se usará un $T_2 \in \Psi\{M_2'\}$, donde M_2' es la matriz básica de M_2 y esta es la comparación de los dígitos 0,1,2,3,4, con 50 imágenes de cada uno, por último se realizara un tercer modelo de regresión multilogística para los números 5,6,7,8,9.

5.2. Analisis de los modelos de regresion

Para el caso de la regresión logística se demostrará que la estimación es capaz de predecir si una imagen nueva pertenece o no a una clase de la muestra. para esto se escoge 100 imágenes del dígito 0 y 100 imágenes del dígito 1, dejando las demás imágenes del 0 y 1 fuera de la muestra para la predicción del modelo. Se implementa la función *glmfit* en matlab, asignando el caso de que la variable respuesta $y_i = 1$ para las imágenes del dígito 1 y $y_i = 0$ para las imágenes del dígito 0.

```
X = muestra([1:200],:);
% las 100 primeras filas son del digito 0 y las otras del digito 1
Y = [zeros(100,1);ones(100,1)]; % asignacion de la respuesta
b = glmfit(X,Y, 'binomial', 'link', 'logit');
% 1 si es Digito 1 y 0 si es el digito 0.
```

De esta manera se obtiene el vector b de los coeficientes β del modelo de regresión, esto da 257 betas diferentes, donde el β_0 es el coeficiente independiente de las variables independientes X.

Ahora se repite el proceso escogiendo un testor típico previamente calculado, es decir ya no se repite el modelo con todos los pixeles de las imágenes sino con los pixeles asignados por el testor típico T_1. Un candidato como testor típico de cardinalidad 14 es:

$$T_1 = \{j_{247},\ j_{249},\ j_{228},\ j_{231},\ j_{245},\ j_{129},\ j_{200},\ j_{243},\ j_{236},\ j_{242},\ j_{135},\ j_{156},\ j_{207},\ j_{237}\}$$

Donde los elementos j_k son columnas de la matriz M', y tiene la información que discriminan los diferentes objetos de varias clases, por lo tanto, estas columnas serán los pixeles (características) más significativos de las imágenes (objetos) al comparar las imágenes del dígito 0 con las imágenes del dígito 1. Se implementa el mismo programa de antes escogiendo las columnas del testor escogido en la muestra de antes.

```
X = muestra([1:200],:);
% las 100 primeras filas son del digito 0 y las otras del digito 1
Y = [zeros(100,1);ones(100,1)]; % asignacion de la respuesta
T = [247 249 228 231 245 129 200 243 236 242 235 156 207 237];
bT = glmfit(muestra([1:200],T),Y,'binomial','link','logit');
```

En este caso se obtiene 15 betas diferentes en lugar de los 257 anteriores, Con los betas calculados se aplica la ecuación 1, en los diferentes modelos obtenidos, para predecir la probabilidad de comparación de nuevas imágenes con respecto la muestra actual, para esta comparación se escoge una imagen que no pertenezca a las 100 primeras imágenes de cada dígito elegido para el cálculo de los betas, esta imagen es escogida de forma aleatoria. En la siguiente tabla se presenta las comparaciones respectivas:

	Imagenes	
	Digito 0	Digito 1
Sin Testor tipico: p	0	1
Con Testor tipico: p	0.0051	0.9866

Tabla 1: Comparacion de la Regresion logistica usando testores tipicos

Para el caso de la regresión multílogística se demostrará que el modelo es capaz de predecir a que clase de la muestra pertenece una nueva imagen de un dígito que no se encuentre en la muestra. Se asigna a los $y_i = 0$ si la imagen es 0, $y_i = 1$ si la imagen es 1, $y_i = 2$ si la imagen es 2 y así respectivamente, en el modelo en matlab siempre se escoge a la última categoría como categoría de referencia. Por limitaciones en materiales de computo se agilita la rapidez del cálculo dividiendo la regresión multilogística en dos partes, se realizará un modelo para los dígitos del 0 al 4 y otro modelo para los dígitos del 5 al 9, por lo tanto, solo se calculará 5 categorías en cada modelo, además se escoge 50 imágenes de cada uno de ellos para la muestra. Dicho esto, se implementa el siguiente programa:

```
cd = categorical([zeros(50,1);ones(50,1);2*ones(50,1);...
                  3*ones(50,1);4*ones(50,1)]);
B = mnrfit(muestra(1:250),cd); %La muestra de los digitos del 0 al 4
```

De esta manera se obtiene la matriz B de (257×4) de los coeficientes del modelo de regresión multilogístico, donde la primera columna son los β_0 de cada $logit\left(\frac{p_{ij}}{p_{ig}}\right)$ respectivamente y cada columna contienen los betas de los $logit\left(\frac{p_{ij}}{p_{ig}}\right)$ respectivos.

Ahora se realiza el mismo procedimiento para la regresión multilogístico con testores típicos previamente calculados, se escoge dos testores típicos de diferente cardinalidad para una comparación entre ellos:

$T_2 = \{j_{247},\ j_{249},\ j_{228},\ j_{231},\ j_{245},\ j_{129},\ j_{200},\ j_{237},\ j_{83},\ j_{93},\ j_{207},\ j_{191},$
$j_{250},\ j_{189},\ j_{241},\ j_{140},\ j_{252},\ j_{121},\ j_{256},\ j_{98},\ j_{244},\ j_{227},\ j_{192},\ j_{229}\}$

$T_3 = \{j_{247},\ j_{249},\ j_{228},\ j_{231},\ j_{245},\ j_{129},\ j_{200},\ j_{237},\ j_{83},\ j_{93},\ j_{207},\ j_{191},$
$j_{250},\ j_{189},\ j_{241},\ j_{140},\ j_{252},\ j_{114},\ j_{58},\ j_{160},\ j_{159}\}$

De cardinalidad 24 y 21 respectivamente, donde los j_k son columnas de las matriz M', y tiene información significativa que discriminan los dígitos 0,1,2,3,4. Se implementa el mismo programa de antes con el testor escogido escogidos, se dará un ejemplo solo para el T_2 entendiendo que el procedimiento es igual para T_3.

```
% asignacion de la respuesta
Y = [zeros(50,1);ones(50,1);2*ones(50,1);3*ones(50,1);4*ones(50,1)];
T2 = [247 249 228 231 245 129 200 237 83 93 207 191 250 189...
     241 140 252 121 256 98 244 227 192 229];
X = muestra([1:250],T2);
Y = categorical(Y);
%es necesario implementar la funcion categorical para usar mnrfit
BT = mnrfit(X,Y);
```

En este caso se obtiene una matriz BT de (25×4) de los coeficientes β del modelo multilogístico con las características del testor típico T_2 y para T_3 se obtiene una matriz de coeficientes de (22×4). Una vez obtenidos los betas se aplican las ecuaciones 3 y 4, para obtener las regresiones multilogisticas de dichos modelos, se escoge de forma aleatoria varias imágenes pertenecientes a diferentes clases que no pertenezcan a la muestra, para que el modelo pueda predecir a que clase pertenece una imagen nueva fuera de la muestra. En la siguiente tabla se muestra esta comparación:

		Imagenes				
		Digito 0	Digito 1	Digito 2	Digito 3	Digito 4
Todas las caracteristicas	$P(y_i = 0) = p_0$	1	0	0	0	0
	$P(y_i = 1) = p_1$	0	1	0	0	0
	$P(y_i = 2) = p_2$	0	0	1	0	0
	$P(y_i = 3) = p_3$	0	0	0	1	0
	$P(y_i = 4) = p_4$	0	0	0	0	1
Características del Testor tipico T_2	$P(y_i = 0) = p_0$	1	0	0	0	0
	$P(y_i = 1) = p_1$	0	0.9961	0.0038	0	0.0001
	$P(y_i = 2) = p_2$	0	0.1239	0.8662	0.006	0.0040
	$P(y_i = 3) = p_3$	0	0.0577	0.0031	0.9392	0
	$P(y_i = 4) = p_4$	0	0.0048	0	0.0001	0.9951
Características del Testor típico T_3	$P(y_i = 0) = p_0$	1	0	0	0	0
	$P(y_i = 1) = p_1$	0	0.7195	0.0002	0.2510	0.0292
	$P(y_i = 2) = p_2$	0	0.149	0.8508	0.0002	0
	$P(y_i = 3) = p_3$	0.0215	0	0.0082	0.9703	0.0001
	$P(y_i = 4) = p_4$	0	0	0	0.0004	0.9996

Tabla 2: Comparación del primer modelo de Regresión multilogística usando testores típicos

De igual manera se implementa la segunda regresión multilogística:

```
cd = categorical([5*ones(50,1);6*ones(50,1);7*ones(50,1);...
                  8*ones(50,1);9*ones(50,1)]);
B = mnrfit(muestra(251:500),cd); %La muestra de los digitos del 5 al 9
```

De esta manera se obtiene la matriz B de (257×4) de los coeficientes del modelo de regresión multilogístico, donde la primera columna son los β_0 de cada $logit\left(\frac{p_{ij}}{p_{ig}}\right)$ respectivamente y cada columna contienen los betas de los $logit\left(\frac{p_{ij}}{p_{ig}}\right)$ respectivos.

Ahora se realiza el mismo procedimiento para dos regresión multilogístico con testores típicos previamente calculados, se escogen dos testores típicos de diferente cardinalidad para una comparación entre ellos:

$T_4 = \{j_{251},\ j_{252},\ j_{247},\ j_{157},\ j_{215},\ j_{249},\ j_{219},\ j_{244},\ j_{227},\ j_{245},\ j_{242},$

$j_{241},\ j_{243},\ j_{197},\ j_{288},\ j_{174},\ j_{56},\ j_{70},\ j_{182},\ j_{202},\ j_{127},\ j_{240},\ j_{148},$

$j_{223},\ j_{248},\ j_{187},\ j_{239},\ j_{207},\ j_{205},\ j_{176},\ j_{80},\ j_{58},\ j_{186},\ j_{112},\ j_{96}\}$

$T_5 = \{j_{251},\ j_{252},\ j_{247},\ j_{157},\ j_{215},\ j_{249},\ j_{219},\ j_{244},\ j_{227},$
$j_{245},\ j_{242},\ j_{241},\ j_{250},\ j_{84},\ j_{74},\ j_{90},\ j_3,\ j_{80},\ j_{216},\ j_{254},\ j_{246},$
$j_{54},\ j_{186},\ j_{156},\ j_{220},\ j_{144},\ j_{61}\}$

De cardinalidad 35 y 27 respectivamente, donde los j_k son columnas de las matrz M', y tiene información significativa que discriminan los dígitos 5,6,7,8,9. Se implementa el mismo programa de antes con los testores escogidos, se dará un ejemplo solo para el T_4 entendiendo que el procedimiento es igual para T_5.

```
% asignacion de la respuesta
Y = [5*ones(50,1);6*ones(50,1);7*ones(50,1);8*ones(50,1);9*ones(50,1)];
T4 = [251 252 247 157 215 249 219 244 227 245 242 241 243 197 228 174 56...
 70 182 202 127 240 148 223 248 187 239 207 205 176 80 58 186 112 96];
X = muestra([251:500],T4);
Y = categorical(Y);
%es necesario implementar la funcion categorical para usar mnrfit
BT = mnrfit(X,Y);
```

En este caso se obtiene una matriz BT de (36×4) de los coeficientes β del modelo multilogístico con las características del testor típico T_4 y para T_5 se obtiene una matriz de coeficientes de (28×4). Una vez obtenidos los betas se aplican las ecuaciones 3 y 4, para obtener las regresiones multilogísticas de dichos modelos, se escoge de forma aleatoria varias imágenes pertenecientes a diferentes clases que no pertenezcan a la muestra, para que el modelo pueda predecir a que clase pertenece una nueva imagen. En la siguiente tabla se muestra esta comparación:

	Imagenes					
		Digito 5	Digito 6	Digito 7	Digito 8	Digito 9
todas las caracteristicas	$P(y_i = 0) = p_0$	1	0	0	0	0
	$P(y_i = 6) = p_1$	0	1	0	0	0
	$P(y_i = 7) = p_2$	0	0	1	0	0
	$P(y_i = 8) = p_3$	0	0	0	1	0
	$P(y_i = 9) = p_4$	0	0	0	0	1
Características del Testor tipico T_4	$P(y_i = 5) = p_0$	0.9748	0.0072	0	0.0066	0.0114
	$P(y_i = 6) = p_1$	0.0873	0.6976	0	0.0603	0.0001
	$P(y_i = 7) = p_2$	0	0.0034	0.9948	0.0018	0
	$P(y_i = 8) = p_3$	0	0	0	0.9839	0.0161
	$P(y_i = 9) = p_4$	0.0001	0	0.0221	0.0318	0.9460
Características del Testor tipico T_5	$P(y_i = 5) = p_0$	0.9852	0	0	0.0141	0.0008
	$P(y_i = 6) = p_1$	0.4330	0.4304	0	0.1310	0.0056
	$P(y_i = 7) = p_2$	0	0	0.6510	0.0595	0.2895
	$P(y_i = 8) = p_3$	0	0	0	0.8578	0.1422
	$P(y_i = 9) = p_4$	0.0002	0	0.0027	0.0729	0.9242

Tabla 3: Comparación de la segunda Regresión multilogística usando testores típicos

Ahora para tener una idea más general de cuan eficientes son estos testores se analiza las 100 imágenes de cada número restantes que no fueron parte de la muestra de los modelos multilogísticos y se realiza un análisis comparando cuantas predicciones son buenas y cuantas son malas.

Primer Modelo Multilogistico						
Imagenes comparativas	T_2		T_3		256 características	
	Correctas	Incorrectas	Correctas	Incorrectas	Correctas	Incorrectas
Digito 0	86	14	82	18	37	63
Digito 1	70	30	73	27	34	66
Digito 2	70	30	73	27	34	66
Digito 3	81	19	81	19	24	76
Digito 4	81	19	75	25	31	69
Porcentaje de predición	0,776		0,768		0,32	

Tabla 4: Comparación de 100 imágenes de cada dígito del primer modelo multilogístico

Segundo Modelo Multilogístico						
Imágenes comparativas	T_4		T_5		256 características	
	Correctas	Incorrectas	Correctas	Incorrectas	Correctas	Incorrectas
Dígito 5	52	48	54	46	46	54
Dígito 6	55	45	67	33	39	61
Dígito 7	77	23	67	33	34	66
Dígito 8	54	46	53	47	38	62
Dígito 9	48	52	54	46	14	86
Tasa de predición	0,572		0,59		0,342	

Tabla 5: Comparación de 100 imágenes de cada dígito del segundo modelo multilogístico

5.3. Limitación de los modelos

Los modelos anteriormente vistos solo pueden predecir si una imagen se parece o una de las clases dentro de la muestra de datos, sin embargo, si se le ingresa una imagen que no se parezca a ninguna categoría del modelo, esta tratará de asignarle un valor a pesar de que la imagen ingresada no se asemeje a ninguna imagen de la muestra. Esto es debido a que cada coeficiente tiene un valor-p que puede decir que tan significativo es el coeficiente en el modelo estadístico, sin embargo, este trabajo no trata de analizar los valores-p de los coeficientes β. Otra solución es crear una categoría que sea para imágenes que no sean iguales a las anteriores y el modelo pueda predecir si la imagen ingresada es parte de las categorías del modelo de regresión logística y multilogística.

6. Conclusiones

Se estableció un algoritmo para hallar la matriz M' en la parte de los testores típicos, este algoritmo está basado en la proposición 2. Si solo se usara la definición de filas básicas se necesitaría de un mayor número de iteraciones para poder predecir si una fila es mayor que otra, así que por la proposición 2 se puede comprobar si dos filas no son comparables y de esta manera se ahorra el análisis de estas filas no comparables, aumentando la eficiencia del algoritmo.

Como se puede ver en las tablas 1 y 2, la aproximación de los modelos usando los testores es muy buena, por lo que se comprueba la teoría de testores típicos, la cual indica que los testores típicos

son capaces de discriminar diferentes objetos de varias clases, en este caso una imagen perteneciente a una clase dentro de varias clases diferentes no pierde mucha información debido a que los errores son pequeños.

Además, se obtiene mayor error en la comparación con la regresión logística debido a que se escogió una muestra más pequeña para los modelos, esto fue necesario por la capacidad del procesamiento de la computadora utilizada para este proyecto, sin embargo, se puede reducir este error aumentando la muestra de 50 imágenes por clase a 100 imágenes por clase, mejorando el modelo de regresión multilogística y obteniendo mayor demora en los cálculos previamente analizados.

Como se puede ver en 3 en la predicción de T_5 hay un error bastante grande al predecir la imagen del dígito 6, sin embargo al usar T_4 el error es mínimo, es puede ser debido que dentro de la muestra de testores típicos existen testores típicos que a pesar de discriminar todos los objetos de diferentes clases, hay testores que les dan más importancia a algunas clases en lugar de otras, por lo que sería interesante investigar cuales son todos los testores típicos más significativos para la discriminación de todas las clases de objetos y como se los encuentran a estos testores típicos.

Dentro de la muestra de imágenes existen dígitos escritos al apuro de una manera indistinguible incluso dentro de sus propias clases, pues eso era parte de la especificación de la muestra obtenida en Semeion Handwritten Digit Data set. Al momento de poner estos dígitos casi indistinguibles en la muestra, generaban una mayor cantidad de testores típicos y también una mayor variación de error entre los modelos de regresión estudiados, esto tiene sentido debido a que estas imágenes casi indistinguibles requieren de más características para ser distinguibles de las demás, y la probabilidad del modelo no será alta debido a que puede incluso confundirse con imágenes de su misma clase, por lo que una parte de los errores ocasionados en el estudio puede ser debido a estas imágenes.

Referencias

[1] Uno, T. (8 de agosto de 2007). SHD: Sparse Hypergraph Dualization, *Program Codes and Instances for Hypergraph Dualization (minimal hitting set enumeration)*, recuperado de http://research.nii.ac.jp/ uno/dualization.html

[2] Brescia, M. (1994). Semeion Handwritten Digit Data Set. *Semeion Research Center of Sciences of Communication* , via Sersale 117, 00128 Rome, Italy. Recuperado de http://archive.ics.uci.edu/ml/datasets/semeion+handwritten+digit

[3] Shirley, P. Marschner, S. (2009). Fundamentals of Computer Graphics. Third Edition. A. K. Peters: Natick, Massachusetts.

[4] Alba, E. Santana, R. (2010). Generación de matrices para evaluar el desempeño de estrategias de búsqueda de testores típicos. *AVANCES EN CIENCIAS E INGENIERIAS*, 2, A30-A35.

[5] Roldán, P. (2015). Economipedia, Modelo de refresión. Recuperado de http://economipedia.com/definiciones/modelo-de-regresion.html

[6] Rencher, A. Schaalje, G. (2008). Linear Models In Statistics. Provo, Utha, United State: A John Wile & Sons, Inc., Publication.

[7] Pando, V. San Martín, R. (2004). Regresión Logística Multinomial. *Sociedad Española de Ciencias Forestales* 18, ISSN: 1575-2410, 323-327. Recuperado de https://dialnet.unirioja.es/servlet/articulo?codigo=2981898

[8] Osorio, D. Ospina, J. Lenis, D. (2009). Planteamiento del modelo logístico multinomial a través de la función can+onica de enlace de la familia exponencial. *Heuristica* 16, 105-115.

[9] Matlab. (2006). MathWorks, Documentation, Generalized linear Regression, recuperado de https://la.mathworks.com/help/stats/glmfit.html#

[10] Bezanson, J. Karpinski, S. Shah, V. Edelman, A. (2013). Text I/O, *Manual The Julia Language*, Recuperado de https://docs.julialang.org/en/stable/stdlib/io-network/ # Text-I/O-1